**统一的响应式CSS框架的研究与设计**

摘要

一方面，自从2005年互联网进入Web 2.0的时代以来，各中类似桌面软件的Web应用大量涌现，网站的前端开发由此发生了翻天覆地的变化。网页不再只是承载单一的文字和图片，各种富媒体让网页的内容更加生动，用户要求更好的交互体验，这些都是依靠前端开发的技术实现的。但在前端开发的工作中普遍存在代码臃肿、维护成本大的问题。

另一方面，由于市场需求与技术的革新，移动互联网发展越来越猛，现在人手一部的智能手机和应用商店里浩如烟海的应用程序都昭示着移动互联网的大时代已经到来。伴随着移动互联网发展到来的是多样化的手持设备如智能手机、平板电脑和其他消费电子产品。用户对网站项目在不同设备的使用体验要求也越来越高，这使得前端工程师的工作更加困难。

面对工作中遇到的前端开发种种问题，本人尝试地提出一个轻巧灵活的解决方案的设想。之后本人研读了W3C相关的Web标准，在此基础上了解了浏览器对HTML和CSS的解析过程。在对HTML和CSS渲染过程有了全局的了解后，又学习了关于Web性能优化的一些方法并分析研究了现有CSS框架（如OOCSS），然后在OOCSS的设计思路上结合自己对前端开发的理解进行扩展得出一套新的CSS设计模式，最后结合团队实际的需求，并使用LESS设计了一个统一的响应式的CSS框架：XSS。

XSS的架构分为服务层与应用层。在XSS中，本人还提出了结构与表现分离、布局与皮肤分离、组件与容器分离三大CSS设计原则。总的来说，XSS轻巧易用，应用此框架，在规范了CSS开发、促进团队成员间合作、提高开发效率的同时，也能容易地满足越来越高的用户体验要求。

关键词：移动互联网；响应式设计 ；CSS框架

（摘要）英文

On the one hand, since the Internet has entered the age of Web 2.0 in 2005, a variety of similar desktop software, Web applications, Web front-end development and great changes have taken place. Web page is no longer carrying single words and images, all sorts of rich media to let web content more vivid, user requirements better interaction experience, these are all rely on the development of the front-end technology implementation.But,the heavy code and cost of maintainance make development of the front-end difficult.

On the other hand, due to market demand and technology innovation, the mobile Internet development is more and more quick.Now the amount of smartphones and applications on the app store indicates the mobile Internet era has arrived. Along with the arrival of mobile Internet development is a variety of handheld devices such as smartphones, tablets and other consumer electronic products. The use of the website project in different devices experience is required more and more high, which in result, the work of front-end engineers become more and more difficult.

In front of the front-end development problems encountered in the work, I try to put forward a lightweight flexible solution.I learned the web standard on W3C, and parsing process of HTML and CSS behind browser. After learning the HTML and CSS rendering process, I learned about how to optimize the web performance.Then I analysis the CSS framework(such as OOCSS).

（未翻译完）

（关键词）英文

mobile internet；Responsive Web Design；CSS Framework

目录

[绪论 3](#_Toc357386675)

[课题设计的来源与意义： 3](#_Toc357386676)

[国内外相关的研究现状 4](#_Toc357386677)

[响应式网页设计 4](#_Toc357386678)

[Web Content Accessibility Guidelines 2.0 5](#_Toc357386679)

[用眼动追踪提升网站可用性 6](#_Toc357386680)

[Web性能黄金法则 7](#_Toc357386681)

[CSS框架 8](#_Toc357386682)

[课题的研究内容 9](#_Toc357386683)

[本论文的写作思路 9](#_Toc357386684)

[第一章 移动互联网是趋势 10](#_Toc357386685)

[第二章 课题关键技术介绍 10](#_Toc357386686)

[CSS 10](#_Toc357386687)

[CSS的基本语法 11](#_Toc357386688)

[CSS选择器 11](#_Toc357386689)

[CSS盒子模型 12](#_Toc357386690)

[HTML 13](#_Toc357386691)

[HTML元素的基本语法 13](#_Toc357386692)

[HTML标签语义化 14](#_Toc357386693)

[LESS 15](#_Toc357386694)

[变量 15](#_Toc357386695)

[混合 15](#_Toc357386696)

[嵌套 16](#_Toc357386697)

[函数与运算 16](#_Toc357386698)

[Media Queries 17](#_Toc357386699)

[通过link方式插入 18](#_Toc357386700)

[xml方式插入 18](#_Toc357386701)

[@import方式插入 18](#_Toc357386702)

[@media方式插入 18](#_Toc357386703)

[第三章 OOCSS 19](#_Toc357386704)

[OOCSS（Object Oriented CSS） 19](#_Toc357386705)

[分离结构与皮肤 19](#_Toc357386706)

[分离容器与内容 20](#_Toc357386707)

[oocss包含4个基本文件和 一些扩展样式。 20](#_Toc357386708)

[OOCSS的对象模型 21](#_Toc357386709)

[OOCSS具有双倍的性能优势： 21](#_Toc357386710)

[OOCSS的局限性： 21](#_Toc357386711)

[OOCSS不能做到真正的响应式设计 21](#_Toc357386712)

[OOCSS不能避免样式污染 22](#_Toc357386713)

[OOCSS的对象扩展机制还不够完善 22](#_Toc357386714)

[第四章 XSS 23](#_Toc357386715)

[XSS的基本介绍 23](#_Toc357386716)

[XSS的目标 23](#_Toc357386717)

[可预见 23](#_Toc357386718)

[可复用 24](#_Toc357386719)

[可维护 24](#_Toc357386720)

[可扩展 24](#_Toc357386721)

[响应式 24](#_Toc357386722)

[XSS的设计原则 25](#_Toc357386723)

[结构与表现分离 25](#_Toc357386724)

[组件与皮肤分离 25](#_Toc357386725)

[组件与容器分离 26](#_Toc357386726)

[XSS的核心文件 26](#_Toc357386727)

[XSS的特点 27](#_Toc357386728)

[服务层与应用层分离。 27](#_Toc357386729)

[增强对象扩展机制。 27](#_Toc357386730)

[兼容HTML5 27](#_Toc357386731)

[简单易用，维护成本低。 27](#_Toc357386732)

[总结 28](#_Toc357386733)

# 绪论

## 课题设计的来源与意义：

自1989年Berners-Lee提出Web的概念以来，网页技术发生了翻天覆地的变化，其用途由最初的纯学术交流，延伸至如今的搜索、邮箱、门户网站、电子商务网站、论坛、SNS、Wiki等，涉及我们的工作、生活、学习和娱乐的方方面面。互联网世界有着成千上百的网页，负责承载和展示信息。近年来，移动互联网发展迅猛，开始重构人们的生活与思维，前端开发机遇与挑战并存。

在用户类型层面上：Web是一个开放的平台，面向所有的人。Web用户可能是残障人士，他们可能有视觉障碍，可能有听觉障碍，可能有肢体障碍，可能有认知和神经障碍；Web用户还可能是非残障人士，他们也许是使用移动手机、也许是Web-TV和信息岗亭的用户，也许是在吵杂环境下使用网站的用户，也许是第二语言访问的用户。

在网络环境层面上：目前，农村地区信息基础设施建设相对落后，宽带网络在农村入户率较低，网络接入条件落后于城镇，城乡之间“数字鸿沟”较大。

在终端设备层面上：近几年内，移动设备快速崛起，移动互联网慢慢进入人们的生活，预计未来5年内移动设备的使用度会超过桌面计算机。所以需要网站不仅要在桌面计算机大尺寸屏幕上可以为用户提供友好的UI和用户体验，同时在小尺寸屏幕上也应该可以提供一致的用户体验。使得用户可以在桌面大屏幕上和移动小屏幕上平滑的切换使用，同时没有任何的不适应感觉。

在前端技术层面上：虽然Web标准自W3C在1994年成立以来一直被致力推广，但被重视并普遍采用的时间至今却不超过6年，而且整个大环境对Web标准的理解还停留在概念层面，对前端规范和前端最佳实践方案还处于摸索阶段。不同的公司，不同的团队，不同的工程师，在不同的项目里，对最佳实践方案有着不同的定义。

在团队合作层面上：随着用户对使用体验的要求不断增加，对网页的表现力的要求也越来越高，从而导致实现代码越来越复杂，着无疑给团队合作带来了麻烦。页面越复杂，对团队合作的要求就越高。如果合作不默契，很可能需要不停的打补丁，最后让代码变得千疮百孔，没有人愿意去维护。

总之，从整个大环境来看，前端开发的工作变得越来越难。实际工作中本人还面临着各种各样的其它问题，特别是在微信这样一个承载着超过3亿用户量的平台。微信采用敏捷开发的方式，什么是敏捷？腾讯广研助理总经理、微信技术总监周颢说，敏捷就是试错法，用最快的迭代速度不断追求卓越。敏捷是一种态度，允许发布前十分钟的变更，并给予产品决策以最大的自由度。大多数互联网产品，最多只能做到一天几个变更，然而，微信可以做到每天20个变更。

微信前端团队还处在一个摸索的阶段，面对这样的环境，我们迫切需要一个精简、强悍的CSS框架，既能够方便团队及个人的开发，又能使开发出来的网站满足用户的体验要求。

## 国内外相关的研究现状

### 响应式网页设计

随着3G的普及，越来越多的人使用手机上网。移动设备正超过桌面设备，成为访问互联网的最常见终端。于是，网页设计师不得不面对一个难题：如何才能在不同大小的设备上呈现同样的网页？

手机的屏幕比较小，宽度通常在600像素以下；PC的屏幕宽度，一般都在1000像素以上（目前主流宽度是1366×768），有的还达到了2000像素。同样的内容，要在大小迥异的屏幕上，都呈现出满意的效果，并不是一件容易的事。

很多网站的解决方法，是为不同的设备提供不同的网页，比如专门提供一个mobile版本，或者iPhone / iPad版本。这样做固然保证了效果，但是比较麻烦，同时要维护好几个版本，而且如果一个网站有多个portal（入口），会大大增加架构设计的复杂度。

于是，很早就有人设想，能不能"一次设计，普遍适用"，让同一张网页自动适应不同大小的屏幕，根据屏幕宽度，自动调整布局（layout）？

2010年，Ethan Marcotte提出了"自适应网页设计"（Responsive Web Design）这个名词，指可以自动识别屏幕宽度、并做出相应调整的网页设计。Ethan Marcotte在“Responsive Web Design”这篇文章中援引了响应式建筑设计的概念。所谓响应式建筑设计就是设计师尝试建造一种使用一些传感器检测周围环境，比如说温度、湿度、光线等等自动进行调整的房子。【1：《A List Apart》Responsive Web Design、Ethan Marcotte、2010年5月25日】现在我们把这个思路延伸到WEB设计领域。我们可以想，为啥我们要为每个用户群各自打造一套设计方案呢？我们太笨了，有没有更智能的做法？和响应式建筑设计一样，web设计也应该做到智能调整。

显然web设计不能使用传感器，这就要更多的抽象思维。好在现在一些概念已经得到实践了，比如流体布局、弹性布局 、帮助页面重新格式化的media queries和脚本等。但是响应式Web设计不仅仅是关于屏幕分辨率自适应以及自动缩放的图片等等，它更像是一种对于设计的全新思维模式。

### Web Content Accessibility Guidelines 2.0

Web Content Accessibility Guidelines (WCAG)是由W3C跨国个人和组织合作，以提供一个Web内容的可访问性共享标准为目的，实现满足个人、组织、和政府的需要的一份官方文档。

本文档介绍了如何使Web内容对于残疾人也具有可访问性。适合所有网站开发人员(页面架构和网站设计师)以及编辑器开发者阅读。本文档的主要目的是促进可访问性的发展，并且遵循这些指南可以使Web内容对所有人更加有用，无论使用什么终端(例如：桌面浏览器、语音浏览器、移动电话、车载个人电脑等)或者在条件限制的情况下使用(例如：嘈杂的环境、过暗或过亮的房间、或者是免提情况等)。遵循本指南也可以帮助人们更快的找到他们想要的信息。这些指南并非建议开发者不使用图片、视频等，而是解释如何让多媒体内容更加具备可访问性，并且有更多的使用者。

Web可访问性意味着残疾人可以使用Web。更具体地说，可访问性意味着残疾人能感知、理解、浏览网站内容，并与之交互。可访问性也有益于其他人，包括由于衰老而能力发生变化的老年人等。【11：W3C、《WAI》Introduction to Web Accessibility】

一直以来，关于网站的可访问性，业界一直存在这样的争论：可访问性是针对残疾人士而言的还是所有人群？

如果断章取义的话，W3C确实只是说到残疾人士，但其实，就Web本身而言，它是一个开放的平台，Web可访问性就是要让所创建的网站对所有用户都可访问，不管用户的生理能力如何、不管用户是以何种方式访问网站。进行可访问性设计有许多不同原因，其中包括残疾人用户的需求、不同的人访问和使用互联网的不同途径和方式。

视障用户包括色盲用户、完全失明用户（盲人）。如果图片不带有相关文字描述，则视障用户在理解图片方面会存在问题。如果图片没有文字描述，看不见图片的盲人用户就无法知道图片表达的是什么。色盲用户在识别设计元素（包括文字）方面也会存在问题，因为色盲用户所能识别的色彩不足以辨别所有的设计元素（包括背景色和页面颜色）。

听障用户在听觉上存在问题。用声音传达的信息无法被听障用户所理解，简单解决方法是提供另外途径的信息传达方式，而不仅仅是声音，例如用文字描述、用图片。

肢体障碍用户经常无法使用鼠标，除非创建网站的导航和输入方式的需求中就考虑残障人士的需求，否则残障人士可能完全无法使用你的网站。

如果网站比较复杂，要想找到我们所想要的信息经常不太容易。如果网站设计的过于复杂、导航不一致、存在让人分心（抓狂）的重复性动画，情况会更加糟糕。这些设计元素会导致认知和神经有障碍的用户的使用问题，甚至会让这些用户完全无法使用网站。

确实，如果我们存在某方面残障，使用互联网是件困难的事情。然而，web可访问性访问不仅仅帮助到残障人士，良好理解和遵循Web可访问性设计，可以让所有用户都受益、更好的服务用户。

Web可访问性设计还可以让通过以下方式使用你的网站的用户受益：

使用移动手机、Web-TV和信息岗亭的用户

低带宽的用户

在吵杂环境下使用网站的用户

容易被“屏幕眩光”伤到眼睛的用户

开车时的用户

低文化水平的用户

第二语言访问的用户（国外用户）

不同学习方式和习惯的用户

处理好Web可访问性，那么网页搜索问题也可以改善。因为拥有良好可访问性的网页对搜索引擎更加透明，文档中结构化的信息使得网页容易被搜索引擎找到并评估，从而建立更加精确的索引，自然也可以得到到更好的搜索结果。

### 用眼动追踪提升网站可用性

眼动（eye movements）实际上包括注视（fixation）与眼跳（saccade）两种最基本的运动。人们在看世界的时候自我感觉视线是连续的，但从眼动记录当中可以明显看到，事实上眼睛的活动是跳跃式的，某些时候是短暂的停顿，称之为“注视”，某些时候是快速的移动，称之为“眼跳”。在眼动结果图中会通过圆圈与线段来表示，这样既可以看到注视又可以看到眼跳的称之为眼动轨迹图（gaze plot）。【2：《眼动研究心理学导论》、闫国利 / 白学军、2012-1】

眼动追踪并不是科学研究中的一个崭新的方法，认知心理学领域的科学家从19世纪起就开始研究眼动追踪，以其发现眼睛的工作原理。到了21世纪，眼动追踪最终发展成为一门实践技术，不仅用于学术研究，还应用在商业研究中。对于网页可用性，我们也可以通过眼动追踪研究准确记录人们使用网页时所注视的屏幕的位置，应用这些数据，我们就能够制定出网页设计的指南，让用户使用更加方便。【10：用眼动追踪提升网站可用性 Jakob Nielsen\ Kara Pernice\冉令华】

1998 年，Jakob Nielsen, Donald Norman and Bruce Tognazzini创立了Nielsen Norman Group (NN/g)，这家公司专门提供以证据为基础的用户体验研究、培训和咨询。2011年Jakob Nielson博士出版了《用眼动追踪提升网站可用性》，本书根据Web在近些年关于眼动追踪技术在网站可用性研究中应用的最新发展状况，并结合NN/g的大量研究成果，详细阐述了如何从可用性的角度来提高Web设计水平，从而使得用户能够更加满意和高效地使用网站来完成各种任务。

### Web性能黄金法则

一次Web应用程序请求，就是从浏览器发出一些参数到你的服务器，然后服务器上的程序对请求进行处理，在省城浏览器可以识别的内容，包括HTML、CSS、Javascript、图片、Flash等，最后由浏览器将这些内容展现给访问者。雅虎实验室的研究表明，对于大多数网站来说，只有不到10%到20%的响应时间是消耗在从Web服务器上下载HTML文档到浏览器中的。其余的80%到90%的时间都是消耗在页面组件的加载与呈现上。【3：《O'Reilly.High.Performance.Web.Sites》、Steve Souders】

内容再丰富的网站，如果慢到无法访问也是毫无意义的；SEO做的再好的网站，如果搜索蜘蛛抓不到也是没有用的；UE设计的再人性化的网站，如果用户连看都看不到也是空谈。

在经过大量研究、实践与测试后，雅虎实验室公布了Web性能优化的最佳实践方式，一直以来，这些方式指导着国内外前端开发工程师对Web性能的优化，更被誉为“黄金法则”。

这14条黄金法则罗列如下：

第一条、尽可能的减少 HTTP 的请求数

第二条、使用CDN(内容分发网络)

第三条、 添加Expire/Cache-Control 头：Add an Expires Header

第四条、启用Gzip压缩：Gzip Components

第五条、将css放在页面最上面 ( Put Stylesheets at the Top)

第六条、将script放在页面最下面 (Put Scripts at the Bottom )

第七条、避免在CSS中使用Expressions (Avoid CSS Expressions )

第八条、把javascript和css都放到外部文件中 (Make JavaScript and CSS External )

第九条、减少DNS查询 (Reduce DNS Lookups)

第十条、压缩 JavaScript 和 CSS (Minify JavaScript )

第十一条、避免重定向 (Avoid Redirects )

第十二条、移除重复的脚本 (Remove Duplicate Scripts )

第十三条、配置实体标签(ETags) (Configure ETags )

第十四条、使 AJAX 缓存 (Make Ajax Cacheable )

### CSS框架

近来，在Web开发中”框架”是一个相当时髦的词。比如JavaScript 框架 YUI、 JQuery和Prototype 都引起广泛的关注，Web应用框架Rails and Dojo 更是引人瞩目，仿佛所有人都使用某种框架来开发自己的网站。

到底什么是框架? Jeff Croft在《A List Apart》的“Frameworks for Designers”一文中提出：

Framework is a set of tools, libraries, conventions, and best practices that attempt to abstract routine tasks into generic modules that can be reused. The goal here is to allow the designer or developer to focus on tasks that are unique to a given project, rather than reinventing the wheel each time around. Generally speaking, this is the approach taken by the aforementioned JavaScript and web application frameworks.

也就是说，框架就是一套包含工具、函数库、约定，以及尝试从常用任务中抽象出可以复用的通用模块，目标是使设计师和开发人员把重点放在任务项目所特有的方面，避免重复开发。编写CSS也是一样，从最初只是定义文字颜色、内容排版，到现在定义所有的表现。CSS框架也渐渐被重视了，因为大家都认识到：从具象的表现中抽出抽象的模块来重复使用，是减少用户下载、方便团队及个人开发最重要的手段。简单地说，CSS框架就是预先准备好的库，旨在帮助开发者使用CSS实现更简单，更符合标准的样式的web页面。

目前业界主流的CSS框架有normalize、G5、Bootstrap、Golden-Grid-System、Semantic GRID SYSTEM等。下图显示了Google Trends上关于CSS Framework的数据曲线，可以看到CSS框架越来越受到大家的重视。



## 课题的研究内容

CSS语言与HTML语言一样，都是一种弱类型的高级语言，源代码被下载到客户端后通过本地浏览器解析，并把CSS样式的效果呈现出来。当我们开发CSS样式的时候，代码排错和维护一直是件令人烦恼的事情，根本原因就在于CSS语言本身的不严谨性，而浏览器对于CSS又有排错能力，只要发现CSS存在语法错误，就会忽略不计，而不影响浏览器的正常工作。这种包容性无形中又增加了开发者的维护难度。另外，一个网站或项目中会存在大量的CSS重复代码，如何发挥CSS自身的特性，避免这种代码的重复，也是开发者应该思考的问题。

另一方面，对于Web的内容，设备厂商、用户以及Web内容的作者都有着不同的需求和期望。Web程序和设备厂商自然是希望能给用户提供一个特殊的功能组合体，但是往往很少Web内容的作者会单独为他们的产品提供内容。用户则更加希望能够在有相似功能的不同设备访问相同的内容，即使那些设备功能有所差别，用户还是希望能够访问到适合的版本。而Web程序的开发者也不想为不同的版本开发多份代码，他们希望的是“一次设计，普遍适用”，尽可能地减少Web程序的建设与维护成本。

本人尝试提出一个统一的响应式CSS基础框架的设想并进行研究与设计。课题的研究内容涉及Web标准、网页响应式设计、网页可访问性、网页可用性、面向对象软件思想、LESS动态样式语言、浏览器工作原理等。

## 本论文的写作思路

绪论 阐述课题设计的来源与意义、国内外相关的研究现状、课题设计的内容以及本论文的写作思路；

1. 介绍目前互联网的现状，市场上终端多样化，屏幕碎片化，移动互联网是趋势；
2. 课题设计关键技术介绍，包括CSS（层叠样式表）、HTML（超文本标记语言）、Media Queries（媒体查询）以及LESS（动态样式表语言）；
3. 分析OOCSS框架，说明现有解决方案仍存在不足之处。
4. 构想一个统一的响应式CSS基础框架，在国内外权威研究基础上，并通过与微信UI团队负责人多次交流，自行研究并实现了基础框架的设计。
5. 对本课题做出总结。

# 第一章 移动互联网是趋势

随着移动设备的兴起，移动互联网的概念也渐渐融入了人们的生活。从中国当前的网络发展趋势看，随着移动终端设备更新换代的速度越来越快，网民对于移动互联网的访问需求也越来越大，中国正处于移动互联网迅猛发展的阶段。

另一方面，以苹果 IOS 和 Google Android 操作系统为主导的智能手机、ipad、智能电视技术的不断更新，极大地带动了连接移动互联网的智能无线终端的爆炸式增长。到 2011 年底，全球智能手机的出厂数目已经超过了台式机的数量。在中国，根据艾瑞咨询发布的报告显示，截止2012第二季度中国智能手机用户数已达到2.90亿人，环比增长15.1%。

然而，在安卓碎片化日益严重的今天，各大手机厂商仍旧我行我素不断的推出各种不同分辨率、不同比例的手机，比如 4.35 寸的 MX2 和 5.5 寸的GALAXY Note II。据显示，目前国内移动设备使用的屏幕分辨率较为分散，且精度偏低。使用最多的是320\*240的QVGA分辨率，占比24.2%。其次是640\*360占比17.5%。 【4：腾讯 2013《移动设备屏幕分辨率分析报告》】

多种数据表示，市场流行着越来越多的终端，加上屏幕分辨率的碎片化，Web前端开发的挑战越来越大。

# 第二章 课题关键技术介绍

## CSS

层叠样式表（Cascading Style Sheets）, 大多数时候简写成CSS， 是一种用来描述HTML、XML（包括各种XML语言如SVG、XHTML）文档表现的样式表（stylesheet）语言。CSS在屏幕、纸、音频或者其它媒体类型中描述了结构化元素怎样被呈现。CSS是开放 web 的核心语言之一，有一个标准化的W3C规范。CSS现在共有4个版本， CSS1现在已经废弃， CSS2.1作为推荐标准。CSS3 现在分成很多较小的模块，正处于标准化的进程中。 CSS4模块的第一份早期草案正在编写和审核中。

### CSS的基本语法

CSS 规则由两个主要的部分构成：选择器，以及一条或多条声明。

|  |
| --- |
| selector {declaration1; declaration2; ... declarationN } |

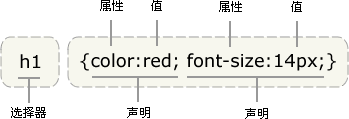
每条声明由一个属性和一个值组成。属性（property）是希望设置的样式属性（style attribute）。每个属性有一个值。属性和值被冒号分开。

|  |
| --- |
| selector {property: value} |

eg: 下面这行代码的作用是将 h1 元素内的文字颜色定义为红色，同时将字体大小设置为 14 像素。

|  |
| --- |
| h1 {color:red; font-size:14px;} |

下面的示意图展示了上面这段代码的结构：



### CSS选择器

选择器是CSS中很重要的概念，所有HTML语言中的标记样式都是通过不同的CSS选择器进行控制的。用户只需通过选择器对不同的HTML标签进行选择，并赋予各种样式声明，就可以实现各种效果，本质上就是一种“内容”与“表现形式”的对应关系。【5：《CSS彻底研究》温谦】

在CSS中，有几种不同类型的选择器，基本选择器有标记选择器、class选择器和ID选择器；通过对基本选择器的重新组合，还可以产生更多种类的选择器，实现更强、更方便的选择功能，复合选择器就是两个或两个以上的基本选择器通过不同的连接方式构成的，复合选择器氛围“交集”选择器、“并集”选择器和后代选择器。CSS2中又引入属性选择器、伪类选择器和伪元素选择器，CSS3中更引入了大量新的选择器。

在工作中，当我们写CSS的时候我们必须注意有些选择器在级联(cascade)上会高于其它选择器，也就是说，我们写在最后面的选择器将不一定会覆盖前面我们写在同一个元素的样式。

我们把特殊性分为4个等级，每个等级代表一类选择器，每个等级的值为其所代表的选择器的个数乘以这一等级的权值，最后把所有等级的值相加得出选择器的特殊值。【9：《CSS那些事儿：掌握网页样式与CSS布局核心技术》林小志 2009 电子工业】

4个等级的定义如下：

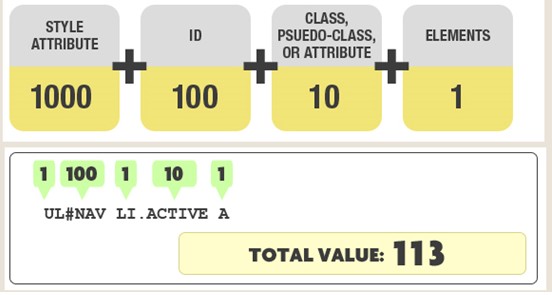
第一等：代表内联样式，如: style=””，权值为1000。

第二等：代表ID选择器，如：#content，权值为100。

第三等：代表类，伪类和属性选择器，如.content，权值为10。

第四等：代表类型选择器和伪元素选择器，如div p，权值为1。

下面给出一个计算CSS权重的例子：



### CSS盒子模型

本质上，CSS中的每个元素都被一个盒子所包围。这个盒子规定了元素框的处理，其中由中心向外扩展包括盒子的尺寸(宽和高：width和height)，内边距（padding）、边框（boader）和外边距（margin）。【6：W3C、《CSS 2.1 Spec》】CSS盒子模型示意图如下所示：



如示意图所示，盒子模型中由width和height规定的区域为实际的内容所能使用的区域；接着，直接包围内容的是内边距（padding），内边距区域内可以显示盒子元素的背景颜色或者背景图片；与内边距接壤的边缘边框（boader）；边框以外是外边距（margin），外边距默认是透明的，即盒子的背景颜色或者背景图片在外边距不可见，它不会遮挡其后的任何元素，但是父元素的内容可透出来可视。

## HTML

超文本标记语言（HyperText Markup Language）简称HTML，是网页的核心语言。你在浏览器里面看到的Web内容都是用HTML描述的。HTML是一种描述Web文档结构和语义的语言。我们可以用<img>、<title>、<p>、<div>等HTML元素对网页进行标记。不同于CSS的是，HTML规范是由W3C和WHATWG一起制定并推行的，随着移动互联网的发展，HTML5应运而生，但HTML5仍处于开发阶段，目前推行的是W3C于1999年12月24日发布的HTML 4.01 。

### HTML元素的基本语法

HTML 文档是由 HTML 元素定义的。HTML 元素指的是从开始标签（start tag）到结束标签（end tag）的所有代码。HTML元素的基本语法列举如下：

HTML 元素以开始标签起始

HTML 元素以结束标签终止

元素的内容是开始标签与结束标签之间的内容

某些 HTML 元素具有空内容（empty content）

空元素在开始标签中进行关闭（以开始标签的结束而结束）

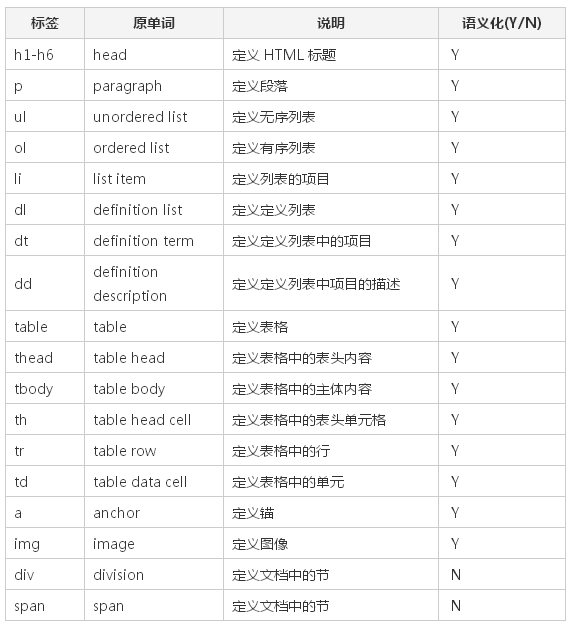
大多数 HTML 元素可拥有属性

eg：下面代码中的 <p> 元素定义了 HTML 文档中的一个段落。这个元素拥有一个开始标签 <p>，以及一个结束标签 </p>。元素内容是：This is my first paragraph。

|  |
| --- |
| <p>This is my first paragraph.</p> |

### HTML标签语义化

语义化是指用合理HTML标记以及其特有的属性去格式化文档内容。语义化的(X)HTML文档有助于提升你的网站对访客的易用性，比如使用PDA、文字浏览器以及残障人士将从中受益。对于搜索引擎或者爬虫软件来说，则有助于它们建立索引，并可能给予一个较高的权值。



以上只列出了常用的一些标签，基本所有的html标签都是一个单词或者词组的缩写，这样其实本意是更便于我们对语义化的理解。其实所有的html标签中，除了<div>和<span>2个无语义的标签，其他标签都有它存在的意义，只有知道有哪些标签，以及对各个标签的本义做一个了解才能知道去用它。

网页重构领域有个节日叫“CSS裸奔节”，WebReBuild.ORG认为，推动Web标准发展应该是每位业界人士的一份责任，因此决定在4月9日号召所有重构人员、站长和关心关注Web标准的朋友，脱去自己网站的外衣，让优雅的层次结构展现给全世界的网民，让语义之美绽放。如果选用的标签几乎全是不带语义的，那么在去样式后网页中几乎看不到任何结构信息，可读性非常差；如果选用的都是语义适合的标签，去样式后网页依然具有非常好的可读性；各个浏览器有自己的默认样式，默认的样式给予了各个标签不同的显示，标签使用的正确与否能体现网站的可用性，这也是检验一个网站可用性的最简单的方法之一。

另一方面，使用语义标签可以确保各种设备以一种有意义的方式来渲染网页。理想情况下，观看设备的任务是符合设备本身的条件来渲染网页。例如,一部手机可以选择使一段标记了标题的文字以粗体显示.而掌上电脑可能会以比较大的字体来显示.无论哪种方式一旦你对文本标记为标题,您就可以确信读取设备将根据其自身的条件来合适地显示页面。

## LESS

Less是一种动态的样式语言。Less扩展了CSS的动态行为，比如说，设置变量（Variables）、混合书写模式（mixins）、运算（operations）和函数（functions）等，而且Less使用了现有的CSS语法。LESS的基本语法如下：

变量

变量允许我们单独定义一系列通用的样式，然后在需要的时候去调用。所以在做全局样式调整的时候我们可能只需要修改几行代码就可以了。

|  |  |
| --- | --- |
| // LESS  @color: #4D926F;  #header {  color: @color;  }  h2 {  color: @color;  } | /\* 生成的 CSS \*/  #header {  color: #4D926F;  }  h2 {  color: #4D926F;  } |

### 混合

混合可以将一个定义好的class A轻松的引入到另一个class B中，从而简单实现class B继承class A中的所有属性。我们还可以带参数地调用，就像使用函数一样。

|  |  |
| --- | --- |
| // LESS  .rounded-corners (@radius: 5px) {  border-radius: @radius;  -webkit-border-radius: @radius;  -moz-border-radius: @radius;  }  #header {  .rounded-corners;  }  #footer {  .rounded-corners(10px);  } | /\* 生成的 CSS \*/  #header {  border-radius: 5px;  -webkit-border-radius: 5px;  -moz-border-radius: 5px;  }  #footer {  border-radius: 10px;  -webkit-border-radius: 10px;  -moz-border-radius: 10px;  } |

### 嵌套

我们可以在一个选择器中嵌套另一个选择器来实现继承，这样很大程度减少了代码量，并且代码看起来更加的清晰。

|  |  |
| --- | --- |
| // LESS  #header {  h1 {  font-size: 26px;  font-weight: bold;  }  p { font-size: 12px;  a { text-decoration: none;  &:hover { border-width: 1px }  }  }  } | /\* 生成的 CSS \*/  #header h1 {  font-size: 26px;  font-weight: bold;  }  #header p {  font-size: 12px;  }  #header p a {  text-decoration: none;  }  #header p a:hover {  border-width: 1px;  } |

### 函数与运算

运算提供了加，减，乘，除操作；我们可以做属性值和颜色的运算，这样就可以实现属性值之间的复杂关系。LESS中的函数一一映射了JavaScript代码，如果你愿意的话可以操作属性值。

|  |  |
| --- | --- |
| // LESS  @the-border: 1px;  @base-color: #111;  @red: #842210;  #header {  color: @base-color \* 3;  border-left: @the-border;  border-right: @the-border \* 2;  }  #footer {  color: @base-color + #003300;  border-color: desaturate(@red, 10%);  } | /\* 生成的 CSS \*/  #header {  color: #333;  border-left: 1px;  border-right: 2px;  }  #footer {  color: #114411;  border-color: #7d2717;  } |

## Media Queries

媒体类型(Media Type)在css2中是一个常见的属性，也是一个非常有用的属性，可以通过媒体类型对不同的设备指定不同的样式，在css2中我们常碰到的就是all（全部），screen（屏幕），print（页面打印或打印预览模式），其实媒体类型不止这三种，W3C定义的所有媒体类型如下表：



页面中引入媒体类型方法也有多种：

### 通过link方式插入

|  |
| --- |
| <link rel="stylesheet" type="text/css" href="../css/print.css" media="print" /> |

### xml方式插入

|  |
| --- |
| <link rel="stylesheet" type="text/css" href="../css/print.css" media="print" /> |

### @import方式插入

@import引入有两种方式，一种是在样式文件中通过@import调用别一个样式文件；另一种方法是在<head></head>中的<style></style>中引入。

|  |  |
| --- | --- |
| // 样式文件中调用  @import url("css/reset.css") screen;  @import url("css/print.css") print; | // <style>中引入  <head>  <style type="text/css">  @import url("css/style.css") all;  </style>  </head> |

### @media方式插入

|  |
| --- |
| @media screen{  选择器{  属性：属性值；  }  } |

在CSS3中引入了 Media Queries的概念，借助可以查询设备的屏幕尺寸颜色等信息，我们就可以根据不同的设备来写CSS，让网页在不同设备上有更好的用户体验。Media Queries的基本语法如下：

|  |
| --- |
| // Media Queries 基本语法  @media screen and (min-width:1024px) and (max-width:1280px){  body{padding:1em;}  } |

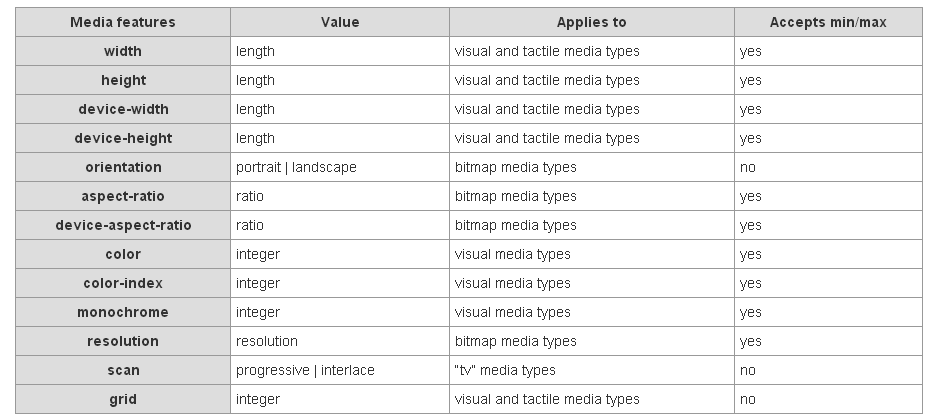
在media属性里：

screen 是媒体类型里的一种，CSS2.1定义了10种媒体类型；

and 被称为关键字，其他关键字还包括 not(排除某种设备)，only(限定某种设备)；

(min-width:1024px) 和（max-width:1280px）就是媒体特性，其被放置在一对圆括号中。

媒体特性共13种，可以说是一个类似CSS属性的集合。但和CSS属性不同的是，媒体特性只接受单个的逻辑表达式作为其值，或者没有值。并且其中的大部分接受 min/max 的前缀，用来表示 大于等于/小于等于 的逻辑，以此避免使用 < 和 > 这些字符。



# 第三章 OOCSS

## OOCSS（Object Oriented CSS）

“面向对象的编程”的概念，开发商之间普遍存在的，他成为任何现代编程语言的一种基本形式，数据的抽象化、模块化和继承等特点在编写代码中得到了大规模的应用。面向对象的CSS是一种容易重用的一种CSS规则，也是OOP的概念，从而降低了页面的加载时间，提高了网面的性能。【7：《Head First Object-Oriented Analysis and Design》、Brett D. McLaughlin / Gary Pollice / Dave West、O'Reilly】OOCSS是Nicole Sullivan在2009年提出来的一套框架，它有两个主要设计原则：

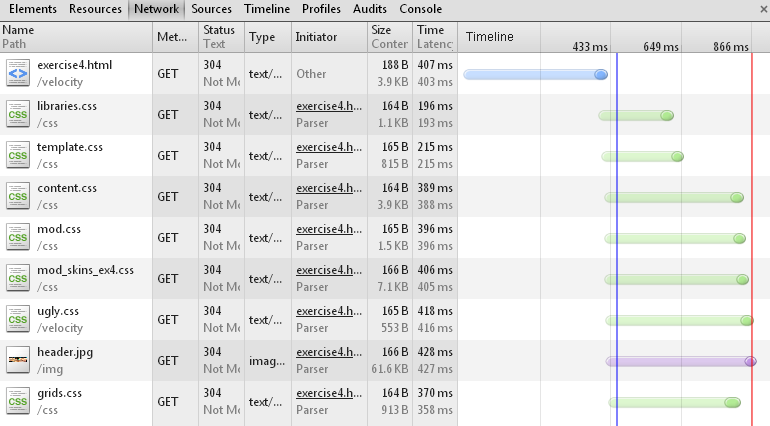
### 分离结构与皮肤

几乎页面上的每个元素都有不同的视觉特效，比如背景、边框、字体等效果，他们在不同的环境中重复的在使用，当把这些视觉效果抽象到一个基于类的模块里，它们就变得可复用，可以运用到任何的元素上，并且具有相同的效果。

### 分离容器与内容

把容器和内容独立出来，这样的好处是，内从插入到任何容器中都可以。但在页面制作中，常常碰到一个组件出现在不同的容器中。对于这样的现象，前端工程师一般都会通过其父元素容器给特定的组件设置不同的样式规则。

## oocss包含4个基本文件和 一些扩展样式。



4个基本文件包括：

libraries.css 用来定义Reset和文字的基本设置。

templete.css 用来定义头部和底部、以及中部的布局。oocss支持最多三栏布局，左右定宽之后中间会自适应。

grids.css 用来定义小范围内的浮动布局。所有的浮动条目都用百分比来定宽，并且最后一个条目.lastunit不设定宽度，这样前面的条目过宽或者过窄的话也不会把最后一条挤下来。

content.css 是对内容和定义的样式。它对所有标题和P都加上padding:10px。

此外还有扩展文件。在这个官方sample中，除了载入上述文件以外，还载入了两个特定的样式文件：mod.css和mod\_skins.css，这两个样式根据特定项目可以自行编写的组件CSS。

## OOCSS的对象模型

OOCSS使用了一个和Java或者PHP语言中对象的概念来描述面向对象的理念，尽管它们的复杂度不一样。OOCSS中的对象由4部分组成：

1. 一个或者多个表示DOM节点的HTML元素
2. 以那个作为包裹节点的类名为开始的所有CSS声明
3. 用于展示背景图片和拼合图片的组件
4. 这个对象相关的 Javascript 交互，媒体或者与之相关的其它方法

下面给出OOCSS所描述的对象原型：

|  |
| --- |
| <div class="mod">  <div class="inner">  <div class="hd"> Mod Head</div>  <div class="bd"> Mod Body</div>  <div class="ft"> Mod Foot</div>  </div>  </div> |

上面代码表示，这个对象是一个模块，并由模块的样式来标示。它包含四部分节点属性，这些不能脱离模块独立存在，包含两个必须的区域：inner 和 body（bd），另两个可选的区域，head（hd）和foot（ft）。

## OOCSS具有双倍的性能优势：

高度重用的CSS代码，只需要很少的CSS代码，意味着：

1、更小的文件，从而更快的传输

2、CSS代码在站点页面中调用的比重增大则有希望被复用或被浏览器缓存

就浏览器而言更少的重绘和布局计算

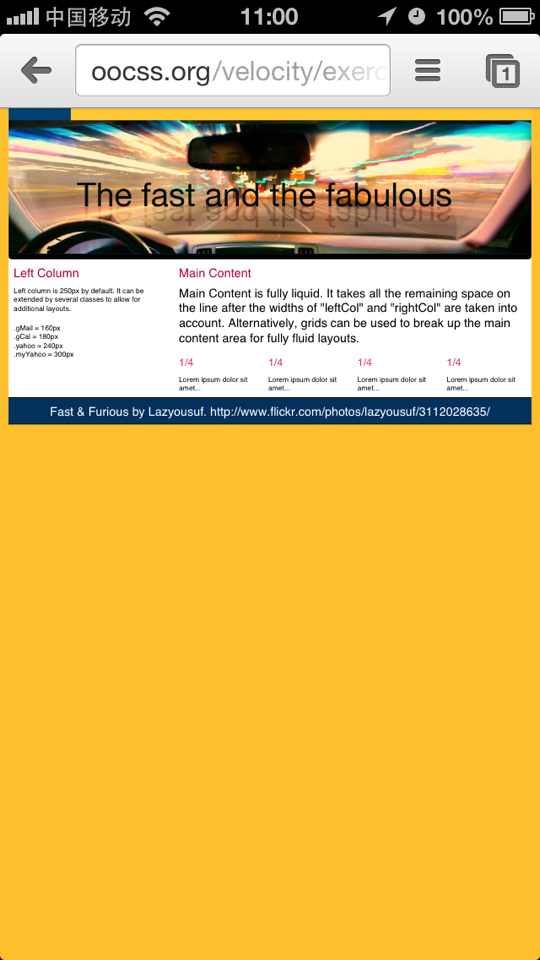
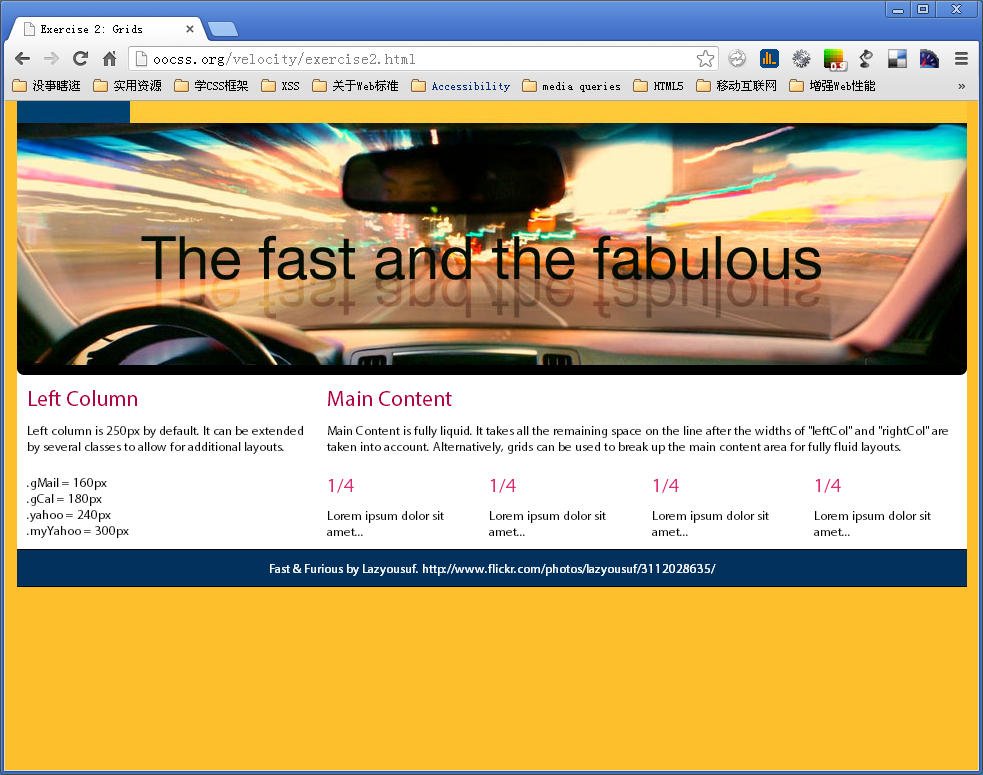
1、单个页面，CSS规则复用的越多，渲染引擎花在“computed values”的计算时间越少

2、手动增加的”extending”类，重写更少的规则，这再一次意味着引擎做很少去应用规则

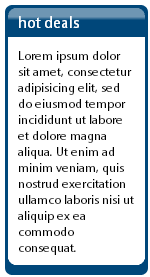
## OOCSS的局限性：

### OOCSS不能做到真正的响应式设计

从OOCSS的grids.css中可以看出所有容器都是用百分比定义宽度的。这就使得网页整体宽度会适配整个屏幕，而页面的元素也将进行等比缩放。下面给出OOCSS Sample在PC、IPad、IPhone浏览器中的显示情况：



### OOCSS不能避免样式污染



这样一个容器，如果用OOCSS实现起来，核心代码如下：

|  |  |
| --- | --- |
| // HTML部分  <div class="mod">  <h2>…</h2>  <p class="content">…</p>  </div> | // CSS部分  .mod{…}  .mod .content{…} |

这样的代码看似没问题，但假如这个容器里面还包含了一个小容器，而这个小容器里面恰好就有一个.content，这个时候小容器的.content也会匹配到上述的代码，但实际上不希望它匹配到大容器的规则，这种情况就往往造成了样式污染。

### OOCSS的对象扩展机制还不够完善

为了让基础对象可重用，我们在不同的地方将其放入不同的容器中，从而通过上下文选择器的使用，让基础对象达到重用的作用。虽然这样表面上达到了重用性，但对于OOCSS来说并不是最理想的方法。在OOCSS中，对象扩展的机制是将基础对象进行扩展，根据上下文语境，通过更多的类名，让基础对象在不同位置实现不同的效果，从而实现了我们前面所说的重用性。简单点说就是通过对基础对象扩展类名，从而达到基础对象的可重用性。

比如一个160px的左列，而非默认值，可以再列上增加额外的class。

|  |  |
| --- | --- |
| // HTML部分  <div class="leftCol gMail"> ... </div> | // CSS部分  .leftCol{float:left;width:250px; }  .gMail{width:160px;} |

但是这样的代码却是很脆弱的，因为很多时候扩展类与基类是有重写关系的，所以它们在CSS文件中的位置关系必须受到约束。假如把.gMail和.leftCol位置颠倒了，那效果就不尽人意了。事实上我们在使用的时候只想关心基类是什么，如何对它进行扩展而已。

# 第四章 XSS

## XSS的基本介绍

为了能更简单的开发、调试和部署，适应需求增长和变化，项目易维护和代码高性能，本人设计出了一个响应式的、轻量级CSS框架，取名XSS。

这个框架是OOCSS的扩展，其实很多方面OOCSS已经做得够好了，但就如上文所言，OOCSS在实际应用中仍存在不足之处，所以有必要对OOCSS进行延伸设计，XSS提出结构与表现分离、布局与皮肤分离、组件与容器分离三大设计原则，旨在建立一套可预见的、可重用的、可维护的和可扩展的CSS设计模式。

特别地，XSS还具备响应式布局的特性。不管是手机屏幕还是PC屏幕还是Pad屏幕，不管分辨率是1280 x 800、1024 x 768、800 x 480还是640 x 960，同样的内容，在这些不同的屏幕上都能呈现出满意的效果。

XSS不单单是CSS文件组合，此框架是基于动态样式表语言LESS进行开发的，基于LESS的开发，可以对组件进行更好的抽象，更好的封装与扩展，也可以增大代码的可复用性，减少开发过程中的代码冗余。对于有复杂交互逻辑的组件，如果使用LESS进行开发会变得更加高效，同时也降低了开发者间的沟通成本。

## XSS的目标

在使用XSS的项目中，CSS是可预见的、可重用的、可维护的和可扩展的、响应式的。

### 可预见

可预见性的CSS就是说我们定义的的规则的最终表现正如你所想，当你添加或更新一条规则，他不应该影响你网站上不想要受影响的部分。对于一个小型网站很少的修改，并不是很重要。但是对于一个有着几十或几百个页面的大型网站，可预见性的CSS是非常必要的。

### 可复用

CSS规范是足够抽象的和耦合的，这样可以根据现有代码部分很快创建出新的组件，而不需要重新编写已经处理过的样式和问题。

### 可维护

当我们的网站需要添加、更新或重新安排一些新的组件和特性，这样做不会重构现有的CSS。给页面添加x组件不会破坏已经存在的组件Y。

### 可扩展

随着网站的规模和复杂程度的增长，它往往需要更多的开发人员来维护。可扩展的CSS意味着可以轻松的由有一个人或一个大型的技术团队管理这个网站。也意味着该网站的CSS架构容易掌握不需要很陡的学习曲线。

### 响应式

移动设备正超过桌面设备，成为访问互联网的最常见终端。可响应的CSS意味着同一个网站，在不同的终端，不管是PC还是Mobile等，都能自动调整布局，让用户获得良好体验。

## XSS的设计原则



### 结构与表现分离

结构与表现分离是语义化HTML的开始，结构指的是HTML标签，HTML本身是没有任何样式的，但是每个浏览器都会有默认样式，尽管这样做的目的也是为了更好的表达html的语义，但是我们不应该依靠这些默认的样式去描述我们的内容。比如Demo中的标题，我们可以使用表达标题含义的标签，如h1、h2、h3、h4、h5、h6。但我们不是因为这些标签默认的加粗效果而用，仅仅是因为它们的本义就是标题而已，所以如果有需要的话还必须在CSS里面显式声明该标签是加粗显示的。

### 组件与皮肤分离

我们经常看到这样的场景，网页上的某个元素的视觉特点在不同的地方有所差异，或者某个元素的视觉特点被重复用在其它元素上。当这些视觉特点抽象成类作为基础的模块时，它们就成为可重复使用，可用于任何元素具有相同的基本结果了。例如，Demo中，我们抽出三套不同的皮肤基类，然后就可以在更多的元素上应用这些皮肤而避免重复定义出现代码冗余的情况了。示例代码如下：

|  |  |  |
| --- | --- | --- |
| // 第一张卡片  <div class="card skin1">  <h2>…</h2>  </div> | // 第二张卡片  <div class=" card skin2">  <h2>…</h2>  </div> | // 第三张卡片  <div class=" card skin2">  <h2>…</h2>  </div> |

### 组件与容器分离

简单地说，就是尽量不要依赖后代选择器达到改变组件的样式。如果从软件工程的角度来讲，也就是“开-闭”原则 （Open-Closed principle, OCP）：一个软件实体应当对扩展开放，对修改关闭。也就是说在设计一个模块的时候，应当使这个模块可以在不被修改的前提下被扩展。

例如，Demo可以抽象成如下模型：

|  |  |  |
| --- | --- | --- |
| // 第一张卡片  <div class="card1">  <h2>…</h2>  </div> | // 第二张卡片  <div class=" card2">  <h2>…</h2>  </div> | // 第三张卡片  <div class=" card3">  <h2>…</h2>  </div> |

假如其中的标题h2就是一个组件，可以看到h2共有两种不同的颜色。所以我们可以把h2抽象出来作为一个基类，然后使用不同的扩展类例如.card1\_title、.card2\_title对它进行扩展。但是不推荐通过.card1 h2，.card2 h2这样的方式对h2进行修改。

首先，这种方式下当h2出现在没有.card1或.card2或.card3的容器中时，它的效果是不可预见的。另外，它也是不可重用的，假如在一个新的.card4的容器中需要一个像.card1里面的h2一样的组件，这时候就必须写多一个关系.card4的声明，这将造成代码冗余。另外，它也比较难维护。一旦这个h2需要重新设计，那么你不得不修改其他几个CSS样式，这样一来也会影响开发效率，如果处理不当的话还会造成样式错乱。

在实际应用中，这三个原则只作为前端团队开发的规范指南，不要求开发者一开始就严格按照XSS设计原则进行CSS开发。理由是，无论我们用什么语言在编写程序，都要做好重构的准备。所谓重构，就是在不改变代码外在行为的前提下，对代码做出修改，以改进程序的内部结构。本质上说，重构就是在代码写好之后改进它的设计。按照目前对软件开发的理解，本应该是先设计而后编码：首先得有一个良好的设计，然后才能开始编码。但是在实际开发中，随着时间流逝，我们不断修改以前的代码，于是根据原先设计所得的系统，整体结构逐渐衰弱。代码质量慢慢沉沦，编码工作从严谨的工程堕落成胡砍乱劈的随性行为。正确的做法是先做，然后优化，不断地进行代码重构，最终才能得出一个真正结构严谨的程序。所以只要保证在进行CSS开发的时候能为接下来的代码重构工作留下扩展空间就可以了。【8：《Refactoring》、Martin Fowler / Kent Beck / John Brant / William Opdyke / Don Roberts、Addison-Wesley Professional、1999-7-8】

## XSS的核心文件

fn.less：这个文件编译后不准产生任何样式，它只是作为一个函数库，里面的函数可以供其它less文件随意调用。

libraries.less：这个文件包含了对浏览器默认表现的重置样式以及所有项目通用的原子类样式。

layout.less：这个文件包含了页面整体布局以及较小范围的栅格化布局。和OOCSS一样，XSS最多支持三栏布局，左右定宽之后中间会自适应。

## XSS的特点

服务层与应用层分离。

在XSS里，服务层指的是fn.less，应用层由公共样式库libraries.less、布局layout.less、组件库mod.less、业务专用样式库demo.less组成。把服务与应用分离使得代码具有更好的规范性、可维护性和更高的重用率，而且在项目开发过程中，也可以同时对服务层与应用层进行开发，加大开发效率。

### 增强对象扩展机制。

在XSS中，利用CSS选择器权重的不同，有效地消除了对象的扩展机制出现的问题。

|  |  |
| --- | --- |
| // HTML部分  <div class="leftCol gMail">  …  </div> | // LESS部分  .leftCol{  float:left;width:250px;  &.gMail{width:160px;}  } |
| // 由LESS生成的CSS  .leftCol{float:left;width:250px; }  .leftCol.gMail{width:160px;} |

### 兼容HTML5

XSS对浏览器默认样式的重置和普通的reset不同，除了一些必要的默认样式重置外，XSS更多的是对HTML5或者其它一些在各主流浏览器里出现不兼容问题的修复，并且通过一些小技巧增加了HTML元素的可用性。

### 简单易用，维护成本低。

除了XSS提供的三个核心文件外，开发者需自行新建一个专门管理可复用组件的LESS文件（例如mod.less）和一个表示某个需求或者某个项目的专用LESS文件（例如demo.less），并且在demo.less中把fn、libraries、layout以及mod这几个文件通过@import方式引用进来。接着就可以依据XSS的三大原则在mod.less中定义页面中的组件了，把特定需求或项目的样式放在demo.less维护就可以了。

# 总结

XSS分离了服务性与应用性的代码，从架构上来说，它分为服务层与应用层。

本质上来说，这个服务层也就是一个通用的变量与函数库，比如一些常见的颜色值变量、CSS盒子模型相关属性的函数、背景与颜色属性的函数、CSS3常用属性的函数等，这些变量与函数都是允许并推荐跨项目调用的，这样一来所有项目的代码就都只存在一个统一的服务型接口，这样一来不仅加强了代码的重复利用，也方便了开发者的代码的管理工作。特别是这个库经过编译后不会产生任何实际代码，几乎不会占用额外的空间。

应用层又分为公用样式库、组件库、布局类库、业务专用样式库。公用样式库提供了一套统一的解决浏览器兼容问题的方案，同时也对浏览器的一些默认样式进行重置并修改以增强HTML元素的可用性。组件库与布局类库也分别提供了统一的组件模型和布局方案。

良好的设计是为了更灵活的应用。XSS轻巧灵活的设计使得应用了XSS的网站更容易适应这个多终端、屏幕分辨率碎片化与越来越高的用户体验要求的环境。

但是，XSS仍存在不足之处。传统的CSS框架中的库与库间没有必然的联系与逻辑关系，虽然XSS服务层与应用层的关系让它这点上变得和传统的CSS框架有所不同，CSS还是一种弱类型的语言，存在代码松散的缺点。另外，虽然XSS是一个通用的框架，但是在实际中不同项目还是会存在一些微小却可能致命的冲突，所以推荐开发者在对应用层有所侧重的修改后再使用。

真正强大的CSS框架应该是在实际开发中不断去提炼和总结，渐进式完善，而并不是一蹴而就的，也不是永远的真理到处适用。XSS目前也只是推出三大设计准则，以及代码的分层观，这些是XSS的核心，在往后的工作中，本人还将也必须继续完善此框架，比如从所有自定义组件类库中抽出一些组件合并到一个公用的组件类，然后与开发同事定义好接口后进行应需加载等，还有关于Web可访问性和可用性方面的技巧，也会越来越多地应用到XSS中。

最后，XSS属开源项目，本人将源代码放在github上托管，写好框架的帮助文档后将邀请一些同行一起维护，希望能为前端开发在中国的发展出一份力。

参考文献